Combining Generative Models and Fisher Kernels for Object Recognition

Alex D. Holub[1], Max Welling[2], Pietro Perona[1]

[1]Computation and Neural Systems
California Institute of Technology, MC 136-93
Pasadena, CA 91125

[2]Department of Computer Science
University of California Irvine
Irvine, CA 92697-3425

Abstract

Learning models for detecting and classifying object categories is a challenging problem in machine vision. While discriminative approaches to learning and classification have, in principle, superior performance, generative approaches provide many useful features, one of which is the ability to naturally establish explicit correspondence between model components and scene features – this, in turn, allows for the handling of missing data and unsupervised learning in clutter. We explore a hybrid generative/discriminative approach using 'Fisher kernels' [1] which retains most of the desirable properties of generative methods, while increasing the classification performance through a discriminative setting. Furthermore, we demonstrate how this kernel framework can be used to combine different types of features and models into a single classifier. Our experiments, conducted on a number of popular benchmarks, show strong performance improvements over the corresponding generative approach and are competitive with the best results reported in the literature.

1 Introduction

Automatically detecting and classifying objects and object categories in images is currently one of the most interesting, useful, and difficult challenges for machine vision. Much progress has been made during the past decade: most significantly in our ability to formulate models that capture the visual and geometrical statistics of natural objects, algorithms that can quickly match these models to images, and learning techniques that can estimate these models from training images with minimal supervision [2, 3, 4, 5, 6, 7, 8]. However, significant challenges remain before we can match human ability

One may divide all learning/classification methods into two broad categories. Call y the label of the class and x the data associated with that class which we can measure: **generative** approaches will estimate the joint probability density function $p(x, y)$ (or, equivalently, $p(x|y)$ and $p(y)$) and will classify using $p(y|x)$ which is obtained using Bayes' rule. Conversely, **discriminative** approaches will estimate $p(y|x)$ (or, equivalently, a classification function $y = f(x)$) directly from the data.

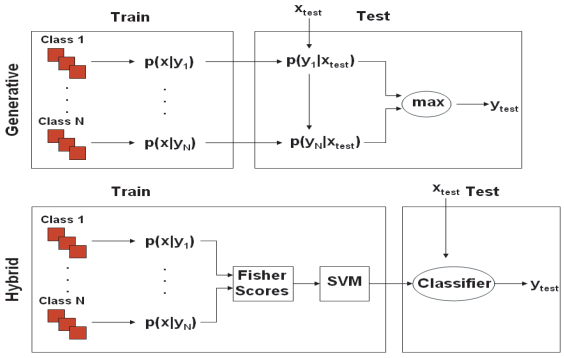

Figure 1: Schematic comparison of the generative and hybrid approaches to learning discussed in this paper.

The prevailing wisdom amongst machine learning researchers is that the discriminative approach is superior: why bother learning the details of the models of different classes if one can learn directly a simpler criterion for discrimination [9]? Indeed, it has been shown that the asymptotic (in the number of training examples) error of discriminative methods is lower than for generative ones [10]. Yet, amongst machine vision researchers generative models remain popular [3, 4, 5, 6, 11, 12].

Generative approaches have a number of attractive properties. First, visual object recognition should be robust to occlusion and missing features. Generative methods provide an intuitive solution to both of these problems by allowing one to establish 'correspondence' between parts of the model and features in the image. This can be accomplished by marginalizing $p(x|y)$ over the missing features and multiplying by the probability of the given pattern of occlusion one may calculate a new pdf $p(x'|y)$ for the observed features x' and generate a new classifier [4]. Second,

collecting training examples is expensive in vision. Ng and Jordan [10] demonstrated both analytically and experimentally that in a 2-class setting the generative approach often has better performance for small numbers of training examples, despite the asymptotic performance being worse. Furthermore, there is some evidence that prior knowledge can be useful [14] within a generative object recognition setting, and generative models tend to easily allow for the incorporation of prior information. In addition, we ultimately envision systems which can learn thousands of categories; in this regime it is unlikely that we will be able to learn discriminative classifiers by considering simultaneously all the training data. It is therefore highly desirable to design classifiers that can learn one category at a time: this is easy in the generative setting and difficult in the discriminative setting. Lastly, very few discriminative approaches have been demonstrated that can learn from examples that contain clutter and occlusion, while this is possible with generative approaches that make explicit hypotheses on the location and structure of the 'signal' in the training examples.

Is it possible to develop hybrid approaches with the flexibility of generative learning and the performance of discriminative methods? Jaakkola and Haussler have recently shown that one can transform a generative approach into a discriminative one by using 'Fisher kernels' [1]. In this paper we show that one can calculate Fisher kernels that are applicable to visual recognition of object categories and explore experimentally their properties on a number of popular and challenging data-sets. Several other kernel-based approaches have been suggested for object recognition [15, 16, 17], including Vasconcelos et al. [17] who exploit a similar paradigm, using a Kullback-Leibler based kernel and test on the COIL data-set.

In section 2 we review the idea of Fisher kernels. In section 3 we develop our generative model and show how Fisher kernels can be applied to these. In section 4 we present experiments. We conclude with a discussion in section 5.

2 Kernel Methods

For supervised learning problems such as regression and classification, kernel methods have proven to be a very successful methodology. As argued in the introduction, our interest is in *combining* generative models with these powerful discriminative tools for the purpose of object recognition. Recognizing that this is a classification task in essence, we have chosen to use support vector machines (SVM) [9] as our kernel machine.

The SVM (like all kernel methods) process the data in the form of a kernel matrix (or Gram matrix) which represents a symmetric and positive definite $n \times n$ matrix of

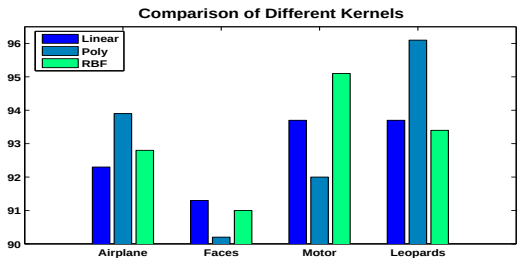

Figure 2: Performance comparison of various kernels on several data-sets. The parameters used to train and test these models are described in the experimental section. The polynomial kernel was of degree 2. The y-axis indicates the classification performance, note that the scale starts at 90%. These results were averaged over 5 experiments. 100 train/test examples used.

similarities between all data-items. A simple way to construct a valid kernel matrix is by defining a set of features, $\boldsymbol{\phi}(x_i)$, and to define the kernel matrix as,

$$K(x_i, x_j) = \boldsymbol{\phi}^T(x_i)\boldsymbol{\phi}(x_j) \tag{1}$$

The generative model will have its impact on the classifier through the definition of these features. In particular, we will follow [1] in using "Fisher scores" as our features. Given a generative probabilistic model they can be extracted as follows,

$$\boldsymbol{\phi}(x_i) = \frac{\partial}{\partial \boldsymbol{\theta}} \log p(x_i|\boldsymbol{\theta}^{\text{MLE}}) \tag{2}$$

where $\boldsymbol{\theta}^{\text{MLE}}$ is the maximum likelihood estimate of the parameters $\boldsymbol{\theta}$. The value of $\boldsymbol{\theta}^{\text{MLE}}$ is determined by balancing the Fisher scores,

$$\sum_i \frac{\partial}{\partial \boldsymbol{\theta}} \log p(x_i|\boldsymbol{\theta}^{\text{MLE}}) = \sum_i \boldsymbol{\phi}(x_i) = 0 \tag{3}$$

Hence, data-items compete to determine the MLE value of the parameters. Two data-items that exert similar forces on all parameters have their feature vectors aligned resulting in a large positive entry in the kernel matrix.

Since it is not a priori evident that the data can be separated using a hyperplane in this feature space it can be beneficial to increase the flexibility of the separating surface (making sure the the problem is properly regularized). This is easily achieved by applying non-linear kernels such as the RBF kernel or the polynomial kernel in this new feature space, i.e. $K(\boldsymbol{\phi}(x_i), \boldsymbol{\phi}(x_j))$ with,

$$K_{\text{RBF}}(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2}||\boldsymbol{\phi}(x_i) - \boldsymbol{\phi}(x_j)||^2\right) \tag{4}$$

$$K_{\text{POL}_p}(x_i, x_j) = (R + \boldsymbol{\phi}(x_i)^T\boldsymbol{\phi}(x_j))^p \tag{5}$$

We mention that from a mathematical point of view it is more elegant to define inner products relative to the inverse Fisher information matrix [1], but we did not see significant performance gains for these classification tasks using the Fisher information matrix.

Given a kernel matrix and a set of labels $\{y_i\}$ for each data-item, the SVM proceeds to learn a classifier of the form,

$$y(x) = \text{sign}\left(\sum_i \alpha_i y_i K(x_i, x)\right) \quad (6)$$

where the coefficients $\{\alpha_i\}$ are determined by solving a constrained quadratic program which aims to maximize the margin between the classes. In our experiments we have used the LIBSVM package freely downloadable from *http://www.csie.ntu.edu.tw/~cjlin/libsvm/*.

Typically, there are a number of design parameters in the problem. These are the free parameters in the definition of the kernel (i.e. σ in the RBF kernel and R, p in the polynomial kernel) and some regularization parameters in the optimization procedure of the $\{\alpha_i\}$. For an SVM the regularization parameter is a constant C determining the tolerance to misclassified data-items in the training set. Values for all design parameters were obtained by cross-validation.

In Figure 2 we compare the performance of the various kernels defined above on two data-sets. In general the performances are similar, although the choice seems to be somewhat dependent on the data-set used. We used RBF kernels for all experiments unless otherwise noted.

2.1 Combining Kernels

There are several situations where we have access to multiple generative models and therefore multiple Fisher scores. For example we may train models with different interest point detectors, varying numbers of parts, describing different aspects of the data (e.g. shape versus appearance) and so on. The question naturally arises how to combine this information into one kernel matrix. The simplest solution is to simply append the Fisher scores into one tall feature vector. Although this is certainly valid, it may not be the optimal choice. For instance, appearance could provide more valuable information for the classification task at hand than shape (or vice versa). A more general approach would be to weight the Fisher scores from the various components differently which translates into linearly combining (with positive coefficients) the corresponding Fisher kernels. Determining these weights is however non-trivial which is the reason we have restricted ourselves to a simple sum of Fisher kernels (with unit weights corresponding to appending Fisher scores) in the experiments reported in sections 4.4 and 4.5.

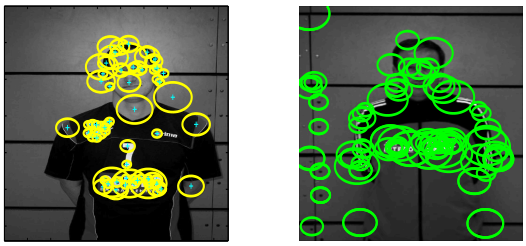

Figure 3: Examples of scaled features found by the KB (left) and multi-scale DoG (right) detectors on images from the 'persons' data-set. Approximately the top 50 most salient detections are shown for both.

3 Generative Models

In this section we briefly describe the generative models which will be used in conjunction with the discriminative methods described above. In principle any differentiable generative model can be used along with the Fisher Kernel. We chose to experiment with a simplified probabilistic Constellation model [4]. We do not explicitly model occlusion or relative scale as done in [5]. Although it is potentially advantageous to include these terms, excluding them allows us to use more features than would be possible in a full model.

3.1 Interest-Point Detection

The constellation model requires the detection of interest points within an image. Numerous algorithms exist for extracting and representing these interest points. We chose to experiment with several popular detectors: the entropy based Kadir and Brady (KB) [18] detector, the multi-scale Difference of Gaussian (DoG) detector [19], the multi-scale hessian detector (mHes), and the multi-scale harris detector(mHar). Figure 3 shows typical interest points found within images. All detectors indicate the saliency of interest points, and only the most salient interest points are used. The locations of the interest points were used to extract 11x11 normalized patches at the scale indicated by the detectors. We typically reduce the dimensionality of the patches to 20 by constructing a PCA basis using features from only the training images and projecting onto that basis. KB was used in all experiments below unless specifically noted.

3.2 The Constellation Model

The constellation model is a generative framework which constructs probabilistic models of object classes by representing the appearance and relative position of several ob-

ject parts. Our goal is to find a set of model parameters θ^{MLE} which maximizes the model. In our model, $\theta = \{\theta_a, \theta_s\}$ represents the means and diagonal variance components of both the appearance and shape models. Consider a set of images belonging to a particular class ranging from $1..N$ and indexed by i. We have extracted both appearance, A_i, and shape, X_i, information from each image I_i using the feature detectors described above. We assume that the shape and appearance models are independent of one another and that the images are I.I.D. The log likelihood of the training images given a particular parameter set θ is:

$$\sum_i \log\left(p(I_i)\right) \quad = \quad \sum_i \log\left(p(A_i|\theta_a) \cdot p(X_i|\theta_s)\right) \quad (7)$$

Next we describe how we solve the correspondence problem, namely the mapping of interest points to model parts. For each I_i we obtain a set of F interest points and their descriptors. We would like to assign a unique interest point to every model component M_j. Since we do not a priori know which interest point belongs to which model component, we introduce a hypothesis variable h which maps interest points to model parts. We order the interest points in ascending order of x-position. We enforce that the positions of all interest points are relative to the left-most interest point, thereby allowing for translational invariance. Note that although we model only the diagonal components of the Gaussian, the model parts are not independent as we enforce that each part is mapped to a unique feature, implicitly introducing dependencies. The result is a total of $\binom{F}{M}$ unique hypotheses, where each h assigns a unique interest point to each model part. We marginalize over the hypothesis variable to obtain the following expression for the log likelihood for a particular class:

$$\sum_i \log\left(p(I_i)\right) = \sum_i \log\left(\sum_h p(A_i, h|\theta_a) p(X_i, h|\theta_s)\right) \quad (8)$$

3.3 Generative Model Optimization

We trained our generative models using the EM algorithm [21]. The algorithm involves iteratively calculating the expected values of the parameters of the model and then maximizing the parameters. The algorithm was terminated after 50 iterations or after the log likelihood stopped increasing. We found empirically that the discriminative performance of the kernel benefitted from keeping the models relatively loose. A 3-part, 25 Feature model took on the order of 20 min to optimize using a combination of optimized Matlab and C (mex) code.

3.4 Fisher Scores for the Constellation Model

In order to train an SVM we require the computation of the Fisher Score for a model. Recall that the Fisher Score is the derivative of log likelihood of the parameters for the model. It is not hard to show that one can compute these derivatives using the following expressions[1],

$$\frac{\partial}{\partial \theta_s} \log\left(p(I_i|\theta)\right) = \sum_h p(h|I_i, \theta) \frac{\partial}{\partial \theta_s} \log p(X_i, h|\theta_s) \quad (9)$$

$$\frac{\partial}{\partial \theta_a} \log\left(p(I_i|\theta)\right) = \sum_h p(h|I_i, \theta) \frac{\partial}{\partial \theta_a} \log p(A_i, h|\theta_a) \quad (10)$$

where both $\{\theta_a, \theta_s\}$ consist of mean and variance parameters of Gaussian appearance and shape models. Despite a potentially variable number of detections in each image I_i its Fisher score has a fixed length. This is because the hypothesis h maps features to a pre-specified number of parts and hence there is a fixed number of parameters in the model.

Most of the execution time of the algorithm occurs during the computation of the Fisher kernels as well as the training of the generative models (20 min for 200 images in a 3-part, 25 Feature model). The SVM training, even with extensive cross-validation, is quite short in comparison due to the relatively small number of training images – on the order of 5 minutes for a set of 200 training images.

4 Experiments

We have performed numerous experiments to determine the efficacy of our technique which we list here: (1) Experiments with the 'Caltech-6' (see, for instance, Fergus et al. [5]). (2) Experiments with few training examples. (3) Experiments training with one background set and testing with another. (4) Experiments using combinations of kernels. (5) Experiments on the Caltech 101 Object Category data-set used by [13, 14]

Some details of the SVM training. Fisher scores were normalized to be within the range [-1,1]. We performed 10x cross-validation to obtain estimates for the optimal values of C. When the RBF kernel was utilized we performed an addition cross-validation to find the optimal value of σ. We varied the cross-validation search space for both parameters on a log base 2 scale from -7,9 in steps of 1 and for C and -8,0 in steps of 2 for σ.

[1] Note that these derivatives are readily available from the EM algorithm at convergence.

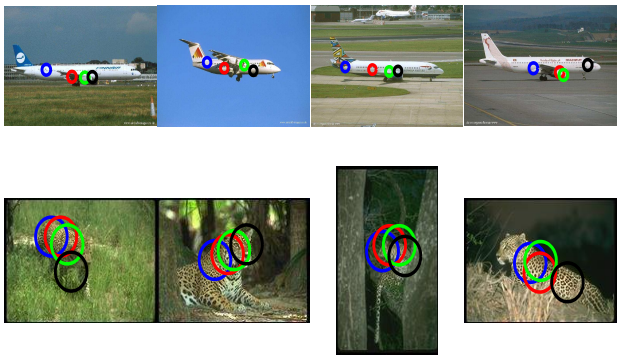

Figure 4: Localization of objects within images using the generative constellation model. Each unique colored circle represents a different part of the model. This is a 4-part model. The positions of the circles represent the hypothesis, h, with the highest likelihood.

Category	Perf	Shape	App	ML	Prev
Faces	**91**	77.7	88.9	83	91.7 [5]
Motorbikes	**95.1**	74.5	91.2	74.2	90.5 [5]
Airplanes	**93.8**	95.3	84.2	72.4	90.8 [5]
Leopards	**93**	71.8	91.3	68.1	91 [5]

Table 1: Performance comparison for Caltech Data-sets. We used 100 training and test images for each class (note that [5] uses far more training images). The background class was the same used by [5]. All scores quoted are the total number correct for both the target class and the background over the total number of examples from both classes. The second column shows the performance of the discriminative algorithm. The third and fourth columns show performance using only the Shape and Appearance Fisher Scores. The Fifth column is the performance using a likelihood ratio on the underlying generative models. The final column shows previous performances on the same data-sets. The underlying generative model contained 3 parts and used a maximum of 30 detected interest points per image. Results were averaged over 5 experiments.

True Class $\Rightarrow$	Motor	Leopards	Faces	Airplanes
Motorcycles	**96.7**	7.3	1.3	.3
Leopards	1.3	**90.7**	.7	0
Faces	1.3	0	**97**	.3
Airplanes	.7	2	1	**99.3**

Table 2: Confusion table for 4 Caltech data-sets. The main diagonal contains the percent correct for each category. Perfect performance would be indicated by 100s along the main diagonal. The classes used in the confusion table are the same used in the generative approach of [5] which achieved performances of 92.5, 90.0, 96.4, and 90.2 across the main diagonal. Using only Fisher scores from the shape and appearance results in 72.6 and 94.8 performance along the main diagonal respectively. 100 training and testing images used. Averaged over 3 experiments.

4.1 Caltech Data-Sets

Table 1 illustrates the performance on the Caltech data-sets[2]. All images were normalized to be the same size. The images contain objects in standardized poses and the categories are visually quite different. In these experiments a single generative model was created of the foreground class from which Fisher scores were extracted for both the foreground and background classes for training. The SVM was trained with the fisher scores from the foreground and background class. Testing was performed using an independent set of images from the foreground and background class by extracting their Fisher scores from the foreground generative model and classifying them using the SVM. We will refer to these experiments as 'class vs. background' experiments henceforth as they involve discrimination tasks between one foreground class and one background class.

There are several interesting points of note: (1) We notice high performance with all classification tasks exhibiting performances above 90%. This high performance is in part due to the stereotyped nature of the image sets which contain very little variation in pose, lighting, and occlusions. Furthermore, the foreground and background classes are visually very dissimilar (see Figures 4 and 6). One caveat to the discriminative approach is that the classifier explicitly utilizes statistical information of the background class. If the background training set is not sufficiently representative of the general class of background images this may lead to overfitting and poor generalization performance. We address this point below in section 4.3. (2) The underlying generative model was relatively weak and hence performed poorly (see the ML column in Table 1). (3) Experiments conducted using only the Shape and Appearance Fisher Scores mostly indicate that the combination of the two is more powerful than either in isolation. Furthermore, these results suggest that the importance of the shape and appearance varies between different classification tasks.

It is not clear how to use the discriminative classifiers in Figure 1 to localize the objects within an image. However, a similar generative approach has been shown to localize objects [5] and we show examples using our generative models in Figure 4.

In addition to class vs. background experiments, we conducted classification experiments using multiple object categories. First a generative model was constructed for all classes of interest. Fisher Scores for both train and test images were obtained. Only the train images were used to create both the SVM classifier and the generative distribution. Since an SVM is inherently a two-class classifier we train the multi-class SVM classifier in a 'one-vs-one' manner. For each pair of classes a distinct classifier was

[2]The data-sets, including the background data-set used here, can be found at: http://www.vision.caltech.edu/html-files/archive.html. The Leopards data-set is from the Corel Data-Base

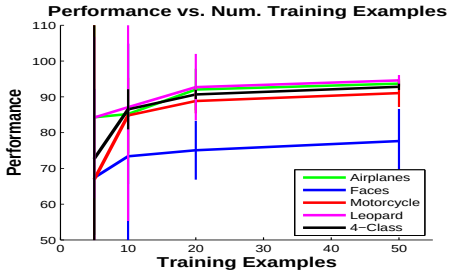

Figure 5: Performance using small numbers of training examples. The x-axis is the number of training examples used for both the foreground and background classes. The y-axis represents the performance. Each unique line represents a different experiment: the black line illustrates the performance of the 4-class experiments, the other lines are for the 'class vs. background' experiments. The variances are represented by the straight perpendicular lines. Note the relatively high initial variances, and the good performance of most models after only 10 training examples (85%+). An RBF kernel was used to train the SVM.

Trained BG ⇒	Caltech1	Caltech2	Graz
Caltech1	**93**	86.5	82
Caltech2	83	**90.5**	78
Graz	83.5	88	**91**
Caltech1	**92**	82	83.5
Caltech2	83	**92.5**	87
Graz	85	85	**91.5**

Table 3: Generalization of different background statistics: Top, Airplane vs. BG experiments. Bottom, Leopards vs. BG experiments. The top row indicates the background data-set trained with. The rows indicate the test set used. The columns indicate the performance of the algorithms. The bold scores indicate the performance on the test examples from the same background class which was trained on, these tend to be the highest performing test sets. We trained and tested with 100 images for each foreground and background category. Results were averaged over 2 experiments. The mHar detector was used.

trained. A test image was assigned to the category containing the largest number of votes among the trained classifiers. All other parameters for training were kept the same as above. Table 2 illustrates a confusion table for 4 Caltech Data-Sets. The discriminative method again outperforms its generative counterpart [5] despite using a much simpler underlying generative model.

4.2 Training with Few Examples

Large numbers of training images are difficult to obtain. For this reason it is useful to explore the performance of recognition algorithms using only small numbers of training images. We constructed a (loose) generative model for the foreground classes using few training examples. Training and testing then proceeded in the same manner as described above. Figure 5 illustrates results on several data-sets. The algorithm seems to performs well in the presence of limited training data.

4.3 Background Classes

Discriminative training for object detection, as employed by [6, 20, 22] among others, implicitly allows the learning algorithm to access the statistics of background data-sets. This differs from generative approaches such as [4] which only minimally utilize the information from the background class during classification. Discriminative algorithms therefore run the risk of over-committing to the statistics of the particular background training images, and hence not generalizing well to arbitrary background images.

We performed several experiments to determine how well different sets of background images generalized to new sets of background images. We considered 3 standard sets of background images: (1) the Caltech background data-set used in [5] (Caltech1), (2) the Caltech background data-set used [14] (Caltech2), (3) the Graz data-set used in [20] (Graz). Images from all three sets are shown in Figure 6. We performed experiments by first generating a model for one foreground class. This generative model was then used to create Fisher scores for the foreground class and a particular background class and an SVM classifier was trained using these scores. We *tested* the classifier on images from all three background classes.

Results for these experiments are summarized in Table 3. This table illustrates that the statistics of a particular background data-set can influence the ability of the classifier to generalize to new sets. These results should be seen as a caveat to using discriminative learning for detection tasks: the statistics of the background images play a crucial role in generalization, especially when relatively few background examples are used.

4.4 Combining Models

We tested the performance of combining multiple kernels using the more challenging Graz data-sets[3], in particular the 'persons' and 'bikes' sets (Figure 7 shows examples from these sets). It is imperative to have large numbers of features when learning on these sets due to the large variability of the objects within the images. We first experimented with combining multiple generative models, with each model containing a different number of parts (2 and 3 parts). Both models were trained on the same data. Fisher

[3]These can be obtained from http://www.emt.tugraz.at/∼pinz/data/

Figure 6: Examples of background images from (top) Caltech 1, (middle) Caltech 2, (bottom) Graz. There are noticeable differences in the image statistics from the different background classes.

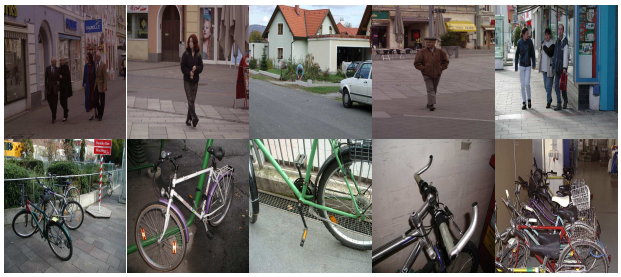

Figure 7: Example images from the Graz persons (top) and bikes (bottom) data-sets. Note the large variations in pose, lighting, occlusion, and scale.

scores were extracted from the foreground generative models on a training set of data for both the foreground and background classes on both 2 and 3 part models and combined into one large vector for SVM training. Test scores were extracted in the same way. Table 4 illustrates results combining simple 2 and 3-part constellation models. The performance training an SVM classifier on Fisher scores for each individual model was less than the combined performance. We anticipate using more features would result in higher performance on the Graz sets and this is an active research area.

In addition we combined models trained using different interest point detectors. Each generative model was trained using different interest points detected on the same set of images. Table 5 shows the results on the same data-sets. There are many more possible combinations of models to explore, and combining models using different kernels is an exciting avenue of future research.

Gen. Model ⇒	Comb	2-Part	3-Part	Prev
persons	**78.5**	75.2	76.4	80.8 [20]
bikes	**75.3**	74.5	74.9	86.5 [20]

Table 4: Effects of combining multiple generative models using the Fisher kernel. Results shown for the Graz 'persons' and 'bikes' sets. 200 images used for training. Note that the combined models outperform individual models. The 2-part and 3-part models used a maximum of 100 and 30 interest points respectively.

Gen. Model ⇒	Comb	KB	DoG	Prev
persons	**73.1**	65.3	77.5	80.8 [20]
bikes	**79.0**	73.3	76.5	86.5 [20]

Table 5: Effects of combining generative models trained using different feature detectors. We used a polynomial degree 2 kernel for experiments.

4.5 Caltech 101

The Caltech 101[4] consists of 101 object categories with varying numbers of examples in each category (from about 30-1000). The challenge of this data-set is to learn representations for many different object classes using a limited number of training examples. However, the depicted objects are mostly well behaved, generally exhibiting relatively small variations in pose, little background clutter, and favorable lighting conditions. Our approach seems well suited for this data-set due both to its strong performance using small numbers of training examples, and the ability to combine different types of information using models from different interest point detectors. In our experiments we created underlying generative models from the categories 'Airplanes' and 'Faces'. We used this to generate Fisher scores for all classes. The SVM was trained using these Fisher scores in a 'one vs one' methodology as described above. For each class we used 20 examples for training and a maximum of 50 for testing. Images were normalized to be the same size for the mHar and mHes detectors.

A reasonable baseline performance was calculated by [13], who found an average performance of 17% using texton histograms. Berg et al. achieve a performance of 45% [13]. The approach of Fei Fei et al. uses an underlying generative model, similar to ours, contained 3 parts, which did not explicitly model occlusion or scale. The algorithm of Fei Fei et al. performs at about 16% on this data-set. Our discriminative (2-part) formulation combining models using the KB, mHar, and mHes detectors results in a performance of 40.1% (classification performance was about 27.1%, 25%, and 29% for the KB, mHar, and mHes detectors individually). A confusion table illustrating our results is shown in Figure 8. We found slightly higher performance

[4] Available at http://www.vision.caltech.edu/html-files/archive.html

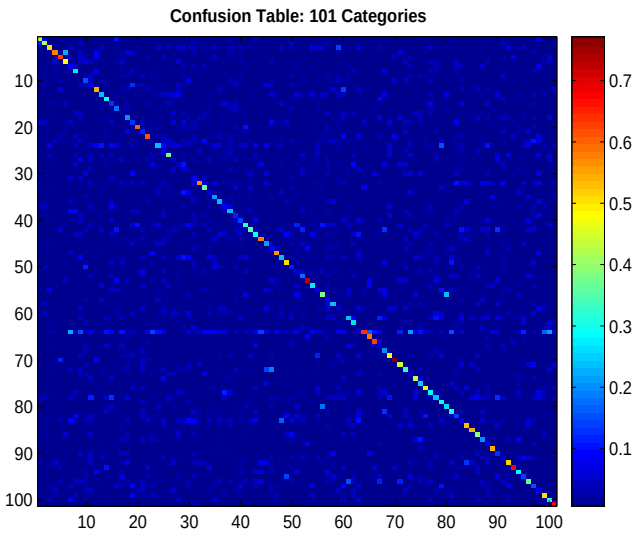

Figure 8: Confusion table for 101 categories. Perfect performance would be indicated by a diagonal line with no-off diagonal colorations. Color-bar shown on right. Performance here was 40.1%. The category labels are the same is in [14]. An RBF kernel was used for training.

using 2-part 100 interest point models compared to 3-part 30 interest point models. 30 PCA coefficients were used. We speculate that the lower performance relative to Berg et al. is, in part, due to the small number of parts and interest points used in our models.

5 Discussion

In this paper we have successfully combined generative models with Fisher kernels to realize performance gains on standard object recognition data-sets. We stress that the formulation can be used with any underlying generative model. Future research includes more rigorous methods for combining kernels and extensions to richer generative models which allow for both more parts and interest point detections.

References

[1] T.S. Jaakkola, D. Haussler. Exploiting generative models in discriminative classifiers. NIPS, 1999. 487-493.

[2] Recognition of Planar Object Classes M.C. Burl, P. Perona CVPR (1996). p. 223

[3] S. Ullman, M. Vidal-Naquet, E. Sali. Visual Features of Intermediate Complexity and their Use in Classification. Nature Neuroscience 5 682-7, 2002.

[4] M. Weber. M. Welling. and P. Perona. Towards automatic discovery of object categories. CVPR, 2000. p. 2101.

[5] R. Fergus, P. Perona, A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. IJCV (2005, submitted).

[6] G. Dorko, C. Schmid. Object class recognition using discriminative local features. Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence.

[7] A. Torralba, K.P. Murphy, W.T. Freeman. Sharing visual features for multiclass and multiview object detection. CVPR, 2004.

[8] A.D. Holub, P. Perona. A Discriminative Framework for Modeling Object Classes. CVPR, 2005.

[9] V. Vapnik. The Nature of Statistical Learning Theory. Springer, N.Y., 1995.

[10] A.Y. Ng, M. Jordan. On Discriminative vs. Generative Classifiers. NIPS 14, 2002.

[11] B. Leibe, B Schiele. Scale-Invariant Object Categorization Using a Scale-Adaptive Mean-Shift Search. DAGM-Symposium 2004: 145-153.

[12] H.Schneiderman. Learning a Restricted Bayesian Network for Object Detection. CVPR, 2004. 639-646

[13] AC. Berg, TL. Berg, J. Malik. Shape Matching and Object Recognition using Low Distortion Correspondences. CVPR (2005).

[14] L. Fei-Fei, R. Fergus. P. Perona. Learning Generative Visual Models from Few Training Examples. CVPR Workshop GMBV, 2004.

[15] C. Wallraven, B. Caputo and A.B.A. Graf. Recognition with Local Features: the Kernel Recipe. ICCV 2003 Proceedings 2, (2003) 257-264

[16] S.Z. Li, Q. Fu, B. Shoelkopf, Y. Cheng, H. Zhang. Kernel machine based learning for multi-view face detection and pose estimation. Proc ICCV, (2001).

[17] N. Vasconcelos, P. Ho, and P. Moreno. The Kullback-Leibler Kernel as a Framework for Discriminant and Localized Representations for Visual Recognition ECCV, 2004. 430-441

[18] T. Kadir, M. Brady. Scale, saliency and image description. IJCV 30(2), 1998. 83-105

[19] J. L. Crowley, A Representation for Shape Based on Peaks and Ridges in the Difference of Low Pass Transform, A. C. Parker, PAMI 6 (2), March 1984.

[20] A. Opelt, M. Fussenegger, A. Pinz and P. Auer. Weak Hypotheses and Boosting for Generic Object Detection and Recognition. ECCV 2004. 71-84

[21] A. Dempster, N. Laird, D. Rubin. Maximum Likelihood from incomplete data via the em algorithm. JRSS B, 39:1-38, 1976.

[22] P. Viola, M. Jones. Robust Real-time Object Detection. ICCV (2001).